Optimizing Distributed System Performance via Adaptive Middleware Load Balancing

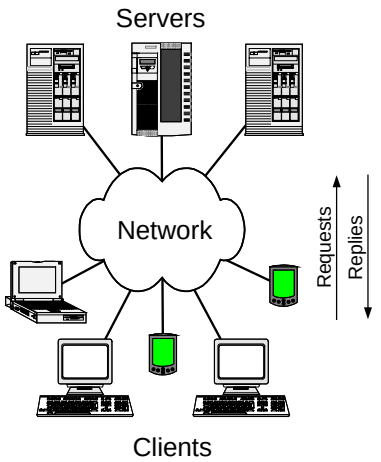
Ossama Othman
ossama@uci.edu

Douglas C. Schmidt
schmidt@uci.edu

Department of Electrical and Computer Engineering
University of California, Irvine
Irvine, California 92697 USA

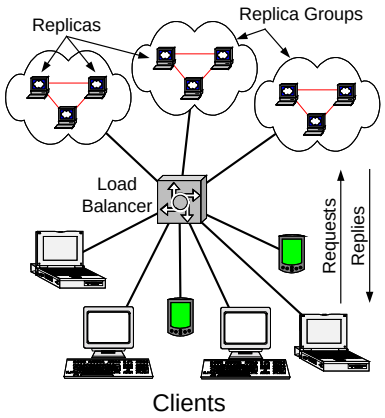
18 June 2001

Introduction



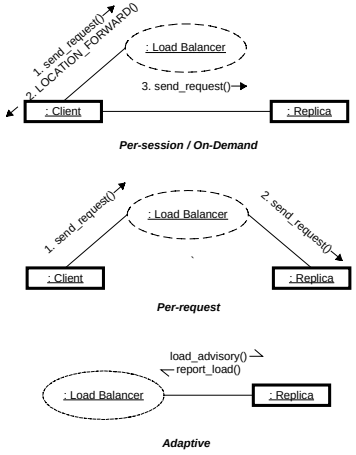
- Motivation
  - Given a resource intensive distributed application
    - Clients typically greatly out number servers
    - Some servers can be more loaded than others
    - Requests generated by clients are often "bursty" and unpredictable
- Solution
  - Adaptive load balancing

Basic Scenario and Concepts



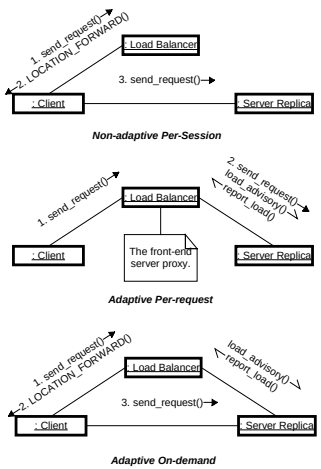
- Load balancing goals
  - Use load balancing to distribute client requests equitably among several *replicas*, within a *replica group*
  - Ensure differences in replica loads are kept to a minimum
- Common Problem
  - Load balancing algorithms in use may be very good but underlying *mechanism* is often inefficient

Load Balancing Strategies



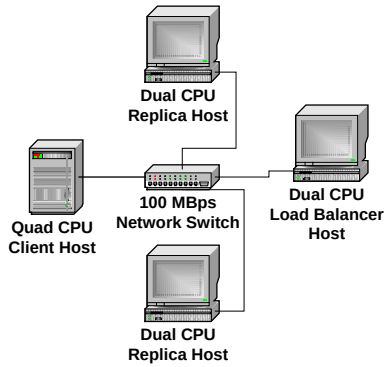
- Client binding granularity
  - Per-session
    - Client permanently forwarded to a replica
  - Per-request
    - Requests forwarded on client's behalf
  - On-demand
    - Client can be rebound to another replica whenever necessary
- Balancing policy
  - Non-adaptive
    - No load feedback used when binding clients
  - Adaptive
    - Load feedback taken in to account

Load Balancing Architectures



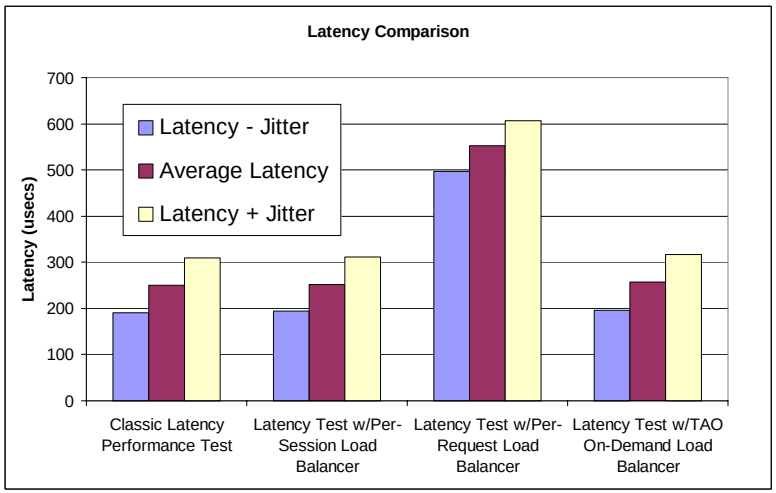
- Load balancing architecture comprised of a combination of *client binding granularity* and *balancing policy*
- Given the strategies just described, there are six possible architectures
- Three common architectures
  - Non-adaptive per-session
  - Adaptive per-request
  - Adaptive on-demand

Load Balancing Experiment Testbed

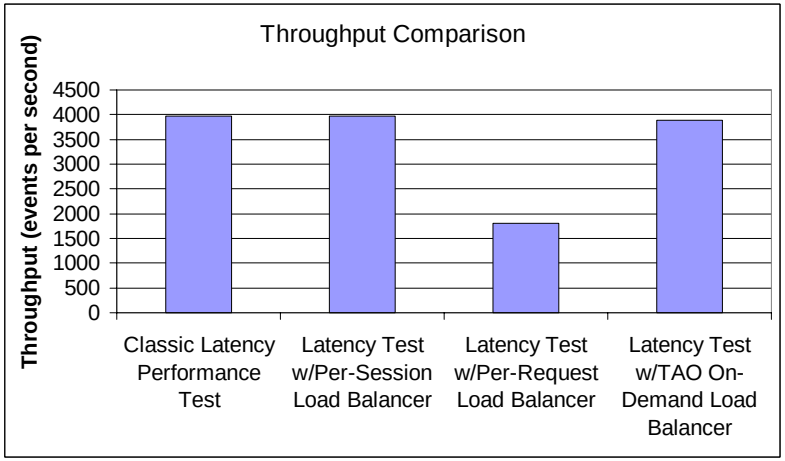


- Testbed hardware
  - Client host (1)
    - Quad CPU 400 MHz Pentium II Xeon, 1GB RAM
  - Replica hosts (2), and load balancer host (1)
    - Dual CPU 733 MHz Pentium III, 512MB RAM
- Testbed software
  - Debian GNU/Linux 2.1 "potato" (glibc 2.1, kernel 2.2.16)
  - TAO "Latency" performance test

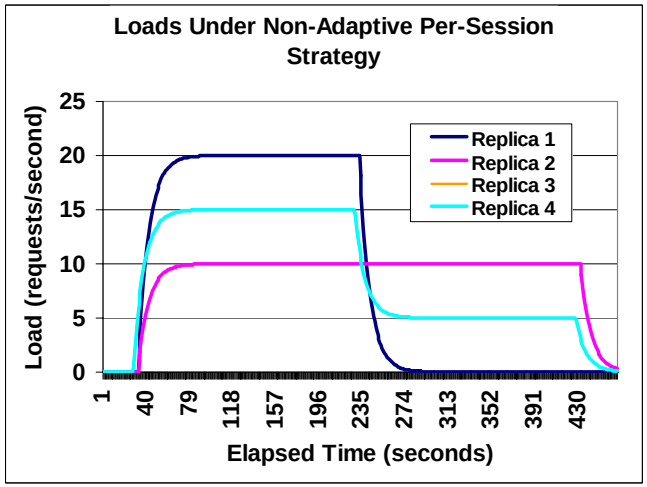
Latency Overhead



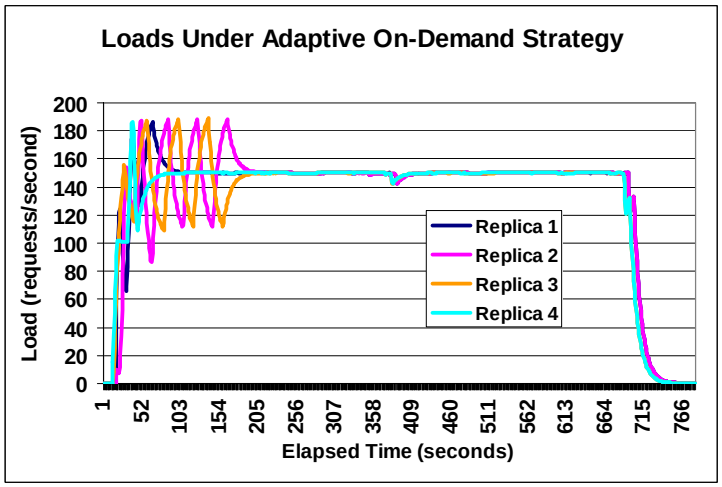
Throughput Overhead



Non-adaptive Per-session Effectiveness



Adaptive On-demand Effectiveness



Conclusion

- Load balancing can be performed at several levels
  - The network level
  - The operating system level
  - The middleware level
- Network-based and OS-based suffer from several limitations
  - Inability to support *application-defined* load metrics at run-time
  - Lack of adaptability due to absence of load feedback, and lack of control over replicas
- Middleware-based load balancing has a clear advantage since it suffers from neither of these limitations